Subject: Re: Attaching PID 0 to a cgroup
Posted by Dhaval Giani on Tue, 01 Jul 2008 09:47:34 GMT
View Forum Message <> Reply to Message

[put in the wrong alias for containers list correcting it.]

On Tue, Jul 01, 2008 at 03:15:45PM +0530, Dhaval Giani wrote:
> Hi Paul,
>
> Attaching PID 0 to a cgroup caused the current task to be attached to
> the cgroup. Looking at the code,
>
> if (pid) {
> rcu_read_lock();
> tsk = find_task_by_vpid(pid);
> if (!tsk || tsk->flags & PF_EXITING) {
> rcu_read_unlock();
> return -ESRCH;
> }
> get_task_struct(tsk);
> rcu_read_unlock();
>
> if ((current->euid) && (current->euid != tsk->uid)
> && (current->euid != tsk->suid)) {
> put_task_struct(tsk);
> return -EACCES;
> }
> } else {
> tsk = current;
> get_task_struct(tsk);
> }
>
> I was wondering, why this was done. It seems to be unexpected behavior.
> Wouldn't something like the following be a better response? (I've used
> EINVAL, but I can change it to ESRCH if that is better.)
>
> ---
> cgroups: Don't allow PID 0 to be attached to a group
>
> Currently when one trys to attach PID 0 to a cgroup, it attaches
> the current task. That is not expected behavior. It should return
> an error instead.
>
> Signed-off-by: Dhaval Giani <dhaval@linux.vnet.ibm.com>
>
> Index: linux-2.6/kernel/cgroup.c
> ===
> --- linux-2.6.orig/kernel/cgroup.c

> +++ linux-2.6/kernel/cgroup.c
> @@ -1309,8 +1309,7 @@ static int attach_task_by_pid(struct cgr
> 	return -EACCES;
> }
> } else {
> - tsk = current;
> - get_task_struct(tsk);
> + return -EINVAL;
> }
>
> ret = cgroup_attach_task(cgrp, tsk);
> --
> regards,
> Dhaval

--
regards,
Dhaval

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup
Posted by Li Zefan on Tue, 01 Jul 2008 10:28:07 GMT
View Forum Message <> Reply to Message

CC: Paul Jackson <pj@sgi.com>

Dhaval Giani wrote:
> [put in the wrong alias for containers list correcting it.]
>
> On Tue, Jul 01, 2008 at 03:15:45PM +0530, Dhaval Giani wrote:
>> Hi Paul,
>>
>> Attaching PID 0 to a cgroup caused the current task to be attached to
>> the cgroup. Looking at the code,
>>

[...]

>>
>> I was wondering, why this was done. It seems to be unexpected behavior.
>> Wouldn't something like the following be a better response? (I've used
>> EINVAL, but I can change it to ESRCH if that is better.)
>>

Why is it unexpected? it follows the behavior of cpuset, so this patch will
break backward compatibility of cpuset.

But it's better to document this.

Document the following cgroup usage:
 # echo 0 > /dev/cgroup/tasks

Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>
---
 cgroups.txt | 4 ++++
 1 file changed, 4 insertions(+)

diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
index 824fc02..213f533 100644
--- a/Documentation/cgroups.txt
+++ b/Documentation/cgroups.txt
@@ -390,6 +390,10 @@ If you have several tasks to attach, you have to do it one after another:
  ...
 # /bin/echo PIDn > tasks

+You can attach the current task by echoing 0:
+
+# /bin/echo 0 > tasks
+
 3. Kernel API
 ==========

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup
Posted by Dhaval Giani on Tue, 01 Jul 2008 10:51:26 GMT
View Forum Message <> Reply to Message

On Tue, Jul 01, 2008 at 06:28:07PM +0800, Li Zefan wrote:
> CC: Paul Jackson <pj@sgi.com>
>
> Dhaval Giani wrote:
> > [put in the wrong alias for containers list correcting it.]
> >

> > On Tue, Jul 01, 2008 at 03:15:45PM +0530, Dhaval Giani wrote:
> >> Hi Paul,
> >>
> >> Attaching PID 0 to a cgroup caused the current task to be attached to
> >> the cgroup. Looking at the code,
> >>
>
> [...]
>
> >>
> >> I was wondering, why this was done. It seems to be unexpected behavior.
> >> Wouldn't something like the following be a better response? (I've used
> >> EINVAL, but I can change it to ESRCH if that is better.)
> >>
>
> Why is it unexpected? it follows the behavior of cpuset, so this patch will
> break backward compatibility of cpuset.

Ah, I was not aware of that. Thanks!

>
> But it's better to document this.
>

Yes please.

> ---
>
> Document the following cgroup usage:
> # echo 0 > /dev/cgroup/tasks
>
> Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>

Acked-by: Dhaval Giani <dhaval@linux.vnet.ibm.com>

> ---
> cgroups.txt | 4 ++++
> 1 file changed, 4 insertions(+)
>
> diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
> index 824fc02..213f533 100644
> --- a/Documentation/cgroups.txt
> +++ b/Documentation/cgroups.txt
> @@ -390,6 +390,10 @@ If you have several tasks to attach, you have to do it one after another:
> ...
> # /bin/echo PIDn > tasks
>

> +You can attach the current task by echoing 0:
> +
> +# /bin/echo 0 > tasks
> +
> 3. Kernel API
> ==========
>
>

--
regards,
Dhaval

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup
Posted by Paul Jackson on Tue, 01 Jul 2008 18:54:09 GMT
View Forum Message <> Reply to Message

> But it's better to document this.

Good idea.

Acked-by: Paul Jackson <pj@sgi.com>

You (Li Zefan) might want to resend this as a patch, in case Andrew
doesn't happen to see this embedded here.

Something like the following:

  Subject: [PATCH] cgroup: document zero pid means current task

  From: Li Zefan <lizf@cn.fujitsu.com>

  Document that a pid of zero(0) can be used to refer to the
  current task when attaching a task to a cgroup, as in the
  following usage:

    # echo 0 > /dev/cgroup/tasks

  This is consistent with existing cpuset behavior.

  Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>
  Acked-by: Dhaval Giani <dhaval@linux.vnet.ibm.com>
  Acked-by: Paul Jackson <pj@sgi.com>

```
 ---
  cgroups.txt |    4 ++++
  1 file changed, 4 insertions(+)

 diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
 index 824fc02..213f533 100644
 --- a/Documentation/cgroups.txt
 +++ b/Documentation/cgroups.txt
 @@ -390,6 +390,10 @@ If you have several tasks to attach, you have to do it one after another:
    ...
  # /bin/echo PIDn > tasks

 +You can attach the current task by echoing 0:
 +
 +# /bin/echo 0 > tasks
 +
  3. Kernel API
  =============


--
                I won't rest till it's the best ...
                Programmer, Linux Scalability
                Paul Jackson <pj@sgi.com> 1.940.382.4214
```

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup
Posted by Paul Menage on Tue, 01 Jul 2008 19:01:27 GMT
View Forum Message <> Reply to Message

On Tue, Jul 1, 2008 at 3:28 AM, Li Zefan <lizf@cn.fujitsu.com> wrote:
>
> Why is it unexpected? it follows the behavior of cpuset, so this patch will
> break backward compatibility of cpuset.

Agreed. I think we want to keep this behaviour.

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

Subject: Re: Attaching PID 0 to a cgroup
Posted by Andrea Righi on Tue, 01 Jul 2008 21:48:31 GMT
View Forum Message <> Reply to Message

Li Zefan wrote:
> CC: Paul Jackson <pj@sgi.com>
>
> Dhaval Giani wrote:
>> [put in the wrong alias for containers list correcting it.]
>>
>> On Tue, Jul 01, 2008 at 03:15:45PM +0530, Dhaval Giani wrote:
>>> Hi Paul,
>>>
>>> Attaching PID 0 to a cgroup caused the current task to be attached to
>>> the cgroup. Looking at the code,
>>>
>
> [...]
>
>>> I was wondering, why this was done. It seems to be unexpected behavior.
>>> Wouldn't something like the following be a better response? (I've used
>>> EINVAL, but I can change it to ESRCH if that is better.)
>>>
>
> Why is it unexpected? it follows the behavior of cpuset, so this patch will
> break backward compatibility of cpuset.
>
> But it's better to document this.
>
> ---
>
> Document the following cgroup usage:
> # echo 0 > /dev/cgroup/tasks
>
> Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>
> ---
> cgroups.txt | 4 ++++
> 1 file changed, 4 insertions(+)
>
> diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
> index 824fc02..213f533 100644
> --- a/Documentation/cgroups.txt
> +++ b/Documentation/cgroups.txt
> @@ -390,6 +390,10 @@ If you have several tasks to attach, you have to do it one after
another:
> ...
> # /bin/echo PIDn > tasks
>
> +You can attach the current task by echoing 0:

> +
> +# /bin/echo 0 > tasks
> +
> 3. Kernel API
> ============

Wouldn't be more meaningful to specify the bash's builtin echo here
even if it doesn't opportunely handle write() errors?

Using /bin/echo would attach /bin/echo itself to the cgroup, that just
exists, so it seems like a kind of noop, isn't it?

-Andrea

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup
Posted by Dhaval Giani on Tue, 01 Jul 2008 21:54:48 GMT
View Forum Message <> Reply to Message

On Tue, Jul 01, 2008 at 11:48:31PM +0200, Andrea Righi wrote:
> Li Zefan wrote:
>> CC: Paul Jackson <pj@sgi.com>
>>
>> Dhaval Giani wrote:
>>> [put in the wrong alias for containers list correcting it.]
>>>
>>> On Tue, Jul 01, 2008 at 03:15:45PM +0530, Dhaval Giani wrote:
>>>> Hi Paul,
>>>>
>>>> Attaching PID 0 to a cgroup caused the current task to be attached to
>>>> the cgroup. Looking at the code,
>>>>
>>
>> [...]
>>
>>>> I was wondering, why this was done. It seems to be unexpected behavior.
>>>> Wouldn't something like the following be a better response? (I've used
>>>> EINVAL, but I can change it to ESRCH if that is better.)
>>>>
>>
>> Why is it unexpected? it follows the behavior of cpuset, so this patch will
>> break backward compatibility of cpuset.
>>
>> But it's better to document this.

>>
>> ---------------------------------------
>>
>> Document the following cgroup usage:
>> # echo 0 > /dev/cgroup/tasks
>>
>> Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>
>> ---
>> cgroups.txt | 4 ++++
>> 1 file changed, 4 insertions(+)
>>
>> diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
>> index 824fc02..213f533 100644
>> --- a/Documentation/cgroups.txt
>> +++ b/Documentation/cgroups.txt
>> @@ -390,6 +390,10 @@ If you have several tasks to attach, you have to do it one after another:
>>   ...
>> # /bin/echo PIDn > tasks
>> +You can attach the current task by echoing 0:
>> +
>> +# /bin/echo 0 > tasks
>> +
>> 3. Kernel API
>> =============
>
> Wouldn't be more meaningful to specify the bash's builtin echo here
> even if it doesn't opportunely handle write() errors?
>
> Using /bin/echo would attach /bin/echo itself to the cgroup, that just
> exists, so it seems like a kind of noop, isn't it?
>

Yes, you are right. this example should use bash's builtin echo.

--
regards,
Dhaval

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup
Posted by Matt Helsley on Thu, 03 Jul 2008 21:59:35 GMT
View Forum Message <> Reply to Message

On Wed, 2008-07-02 at 03:24 +0530, Dhaval Giani wrote:
> On Tue, Jul 01, 2008 at 11:48:31PM +0200, Andrea Righi wrote:
> > Li Zefan wrote:
> >> CC: Paul Jackson <pj@sgi.com>
> >>
> >> Dhaval Giani wrote:
> >>> [put in the wrong alias for containers list correcting it.]
> >>>
> >>> On Tue, Jul 01, 2008 at 03:15:45PM +0530, Dhaval Giani wrote:
> >>>> Hi Paul,
> >>>>
> >>>> Attaching PID 0 to a cgroup caused the current task to be attached to
> >>>> the cgroup. Looking at the code,
> >>>>
> >>
> >> [...]
> >>
> >>>> I was wondering, why this was done. It seems to be unexpected behavior.
> >>>> Wouldn't something like the following be a better response? (I've used
> >>>> EINVAL, but I can change it to ESRCH if that is better.)
> >>>>
> >>
> >> Why is it unexpected? it follows the behavior of cpuset, so this patch will
> >> break backward compatibility of cpuset.
> >>
> >> But it's better to document this.
> >>
> >> ---
> >>
> >> Document the following cgroup usage:
> >> # echo 0 > /dev/cgroup/tasks
> >>
> >> Signed-off-by: Li Zefan <lizf@cn.fujitsu.com>
> >> ---
> >> cgroups.txt | 4 ++++
> >> 1 file changed, 4 insertions(+)
> >>
> >> diff --git a/Documentation/cgroups.txt b/Documentation/cgroups.txt
> >> index 824fc02..213f533 100644
> >> --- a/Documentation/cgroups.txt
> >> +++ b/Documentation/cgroups.txt
> >> @@ -390,6 +390,10 @@ If you have several tasks to attach, you have to do it one after another:
> >>   ...
> >> # /bin/echo PIDn > tasks
> >> +You can attach the current task by echoing 0:
> >> +
> >> +# /bin/echo 0 > tasks

> >> +
> >> 3. Kernel API
> >> ==========
> >
> > Wouldn't be more meaningful to specify the bash's builtin echo here
> > even if it doesn't opportunely handle write() errors?
> >
> > Using /bin/echo would attach /bin/echo itself to the cgroup, that just
> > exists, so it seems like a kind of noop, isn't it?
> >
>
> Yes, you are right. this example should use bash's builtin echo.

 IMHO you need to include this point in the docs verbosely rather than
just switching the docs to bash's builin-in echo. Otherwise it doesn't
fully resolve the fundamental confusion you correctly identified.

Or perhaps a snippet of simplified C code will make it clear:
```
------------
 char buffer[16];
 int fd;

 fd = open("/some/cgroup/tasks", O_WRONLY);

 /*
  * These two writes produce the same effect: adding this process
  * to /some/cgroup.
  */
 if (the_slightly_shorter_way)
  write(fd, "0", 2);
 else {
  /* The slightly-less-short way */
  snprintf(buffer, 16, "%u", getpid());
  write(fd, buffer, strlen(buffer));
 }
------------
```

Cheers,
 -Matt Helsley

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers

---

Subject: Re: Attaching PID 0 to a cgroup

On Thu, Jul 3, 2008 at 2:59 PM, Matt Helsley <matthltc@us.ibm.com> wrote:
> ------------
> char buffer[16];
> int fd;
>
> fd = open("/some/cgroup/tasks", O_WRONLY);
>
> /*
> * These two writes produce the same effect: adding this process
> * to /some/cgroup.
> */
> if (the_slightly_shorter_way)
> write(fd, "0", 2);
> else {
> /* The slightly-less-short way */
> snprintf(buffer, 16, "%u", getpid());
> write(fd, buffer, strlen(buffer));

If it's a threaded application, then you'd need gettid() rather than
getpid() for the two to be equivalent.

Paul

_____
Containers mailing list
Containers@lists.linux-foundation.org
https://lists.linux-foundation.org/mailman/listinfo/containers